Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра автоматики та управління в технічних системах

Лабораторна робота №4

Архітектура комп’ютера

«ДОСЛІДЖЕННЯ РОБОТИ СТЕКУ МПС В АРХІТЕКТУРІ IA-32 (X86) У REAL ADDRESS MODE»

Виконали:
студенти групи ІТ-01:

Бардін В.Д.
Задніпрянець А.А.
Куркін О.О.

Перевірив:
Бердник Ю. М.

Дата здачі 08.04.2021

Захищено з балом _____

Київ 2021

Вихідний код:

```
TITLE ЛР_4
;-------------------------------------------------------------------------------------------------------------------------
;ЛР  №4
;-------------------------------------------------------------------------------------------------------------------------
; Комп'ютерна архітектура
; ВУЗ:          НТУУ "КПІ"
; Факультет:    ФІОТ
; Курс:          1
; Група:        ІТ-01
;-------------------------------------------------------------------------------------------------------------------------
; Автор:        Бардін В. Д.
;               Задніпрянець А. А.
;               Куркін О. О.
;
; Дата:         04/04/21
;-------------------------------------------------------------------------------------------------------------------------
IDEAL           ; Директива - тип Асемблера tasm
MODEL small     ; Директива - тип моделі пам’яті
STACK 1024      ; Директива - розмір стеку

DATASEG
array_stack dw 2E2Eh, 2098h, 6847h, 7230h, 5182h, 1964h, 7350h, 1628h, 9823h, 5927h, 2343h, 7515h, 5238h, 8272h, 6576h, 2619h
            dw 9520h, 6608h, 5535h, 9745h, 8314h, 1683h, 2065h, 3718h, 3198h, 5267h, 1546h, 2273h, 9317h, 2181h, 7466h, 1940h
            dw 7417h, 2385h, 2240h, 8346h, 9246h, 9573h, 5050h, 1549h, 4402h, 7854h, 8126h, 9060h, 3476h, 7497h, 3703h, 1857h
            dw 8794h, 8017h, 3227h, 1033h, 7980h, 8658h, 6475h, 2653h, 4970h, 3343h, 3788h, 4600h, 4953h, 5156h, 7128h, 9539h
            dw 4889h, 8734h, 8685h, 3104h, 5514h, 9721h, 5958h, 4611h, 7759h, 1725h, 6059h, 7499h, 9681h, 4573h, 6929h, 1387h
            dw 2772h, 5484h, 7392h, 8250h, 5503h, 6080h, 1249h, 3413h, 3167h, 9250h, 7496h, 7533h, 2101h, 4007h, 6810h, 4531h
            dw 4979h, 9050h, 8589h, 6962h, 5374h, 3451h, 2971h, 6612h, 8002h, 4074h, 2634h, 3694h, 7979h, 7571h, 7333h, 9852h
            dw 9847h, 3270h, 2426h, 6722h, 2697h, 9765h, 2203h, 3222h, 6819h, 5024h, 5846h, 3619h, 9626h, 2638h, 4465h, 2E2Eh

first_birthdate db "8.10|09.10|24.04"

CODESEG
Start:
    mov ax, @data ; data segment init
    mov ds, ax
    mov es, ax

; set params for copy_array
    mov cx, 128                 ; repeats amount
    call copy_arr

; set params for array_to_stack
    mov cx, 128                 ; repeats amount
    call array_to_stack

; set params for set_bdates
    mov cx, 16                  ; repeats amount
    mov bp, 0150h               ; offset for birthdates
    call set_bdate

; application finishing
    mov ah, 4ch
    int 21h

;-----------------------------------------------------Copy array procedure-----------------------------------------------
; Input params: cx - initial array size,
; Output params: array copy placed at ds,
;-------------------------------------------------------------------------------------------------------------------------
    PROC copy_arr
        xor si, si                      ; set si to zero
        array_coping_loop:
            mov bx, [ds:si]             ; get number from array_array stack & set it to bx as a temp variable
            mov [ds:[si+260h]], bx      ; set value from bx to ds with offset
            add si, 2                   ; si value + 2
            loop array_coping_loop

        ret
    ENDP

;-------------------------------------------------Put array to stack procedure-------------------------------------------
; Input params:  cx - initial array size,
; Output params: array copy placed at ds,
;-------------------------------------------------------------------------------------------------------------------------
    PROC array_to_stack
        xor si, si                      ; set si to zero
        push_to_stack:
            mov ax,[ds:[si]]            ; set value to ax from ds with offset si
            push ax                     ; push ax value to stack
            add si, 2                   ; si + 2
            loop push_to_stack

        ret
    ENDP

;-----------------------------------------------Add birthdate to stack---------------------------------------------------
; Вхідні  параметри: cx - symbols amount at birthday,
;                    bp - offset
; Вихідні параметри: array copy placed at ds,
;                    array begin address at ax
;-------------------------------------------------------------------------------------------------------------------------
    PROC set_bdates
        xor si,si                               ; set si to zero
        birthdate_label:
            mov ah, [first_birthdate+si]        ; set value to ah from first_birthdate with offset si
            mov [bp], ah                        ; add value to stack
            inc si                              ; increment si
            inc bp                              ; increment bp
            loop birthdate_label

        ret
    ENDP
end Start
```

Результати дослідження: Було створено двовимірний масив, що складається з елементів у два байти (слово) кожен, має розмір 16x8 (для зручності виведення). Потім за допомогою процедури `copy_arr` було зроблено копію масиву.

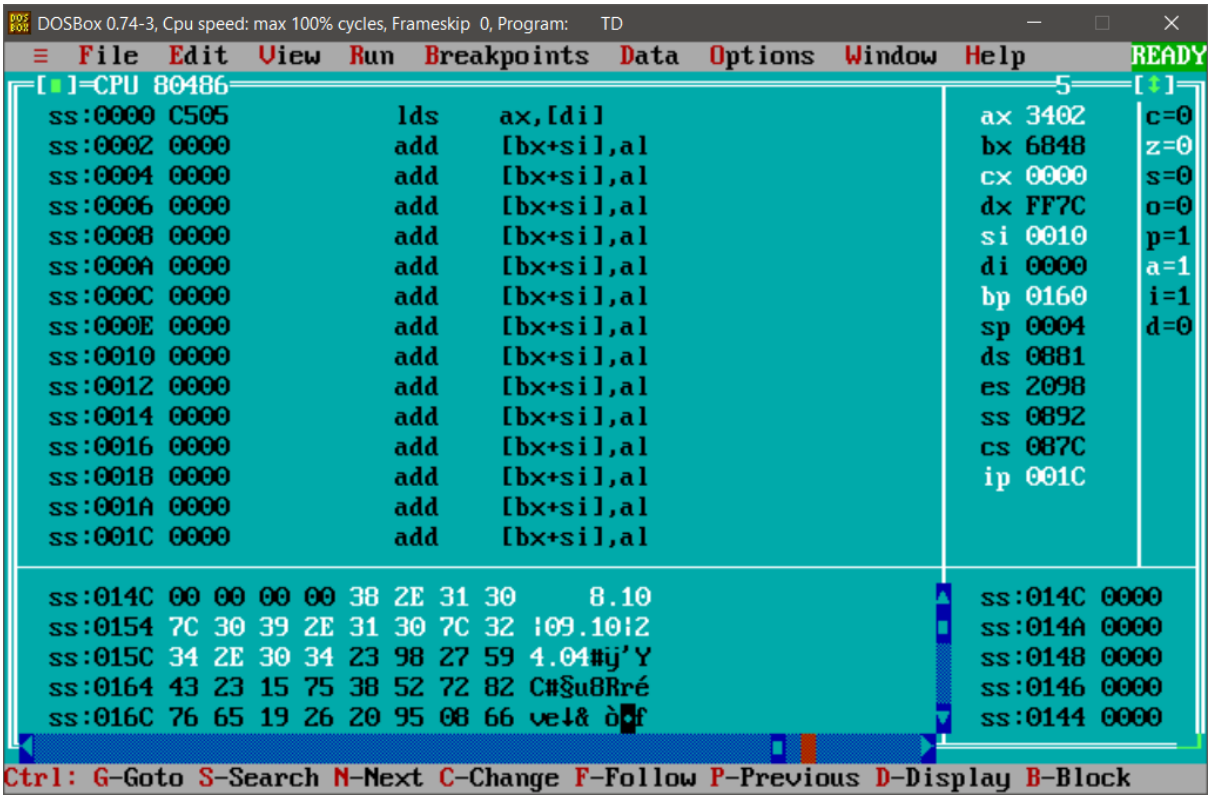
```
;------------Copy array procedure--------------
; Input params: cx - initial array size,
; Output params: array copy placed at ds,
;----------------------------------------------
    PROC copy_arr
        xor si, si                      ; set si to zero
        array_coping_loop:
            mov bx, [ds:si]             ; get number from array & set it to bx
            mov [ds:[si+260h]], bx      ; set value from bx to ds with offset
            add si, 2                   ; si value + 2
            loop array_coping_loop

        ret
    ENDP
```

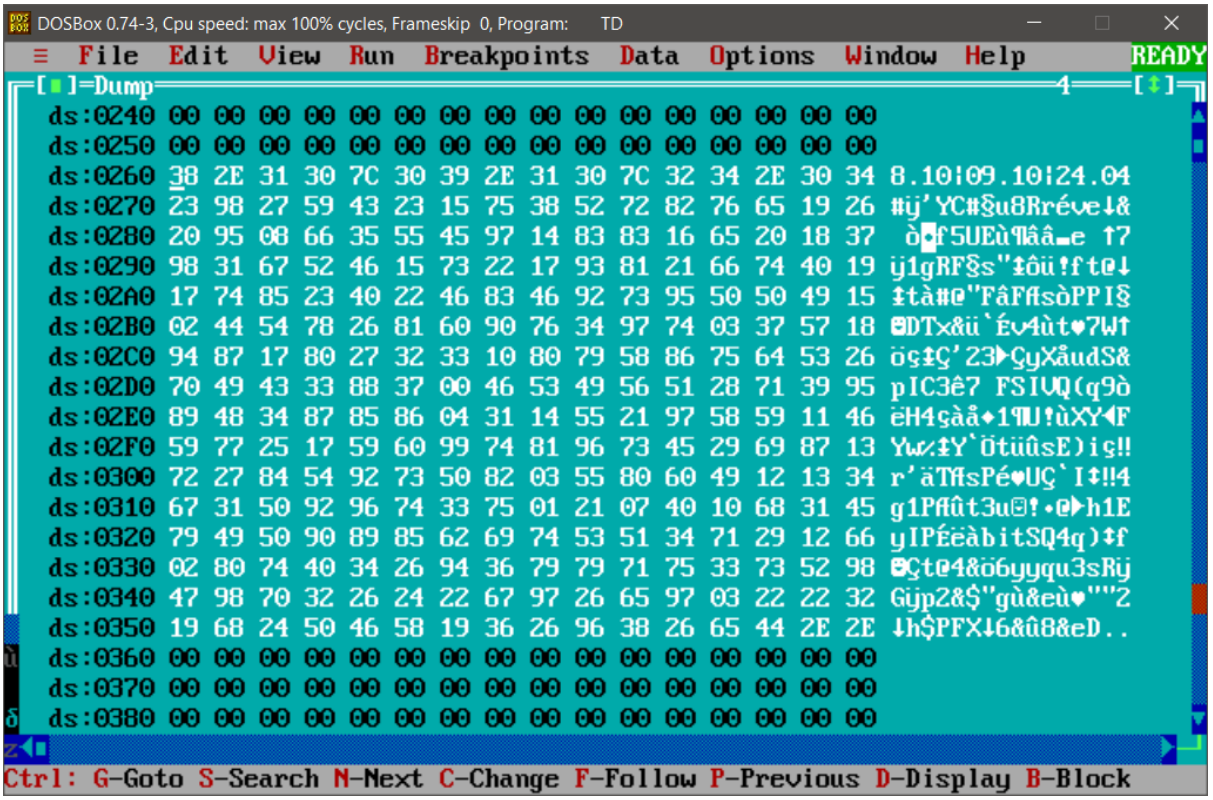
Далі за допомогою процедури `array_to_stack` помістили копію і стек.

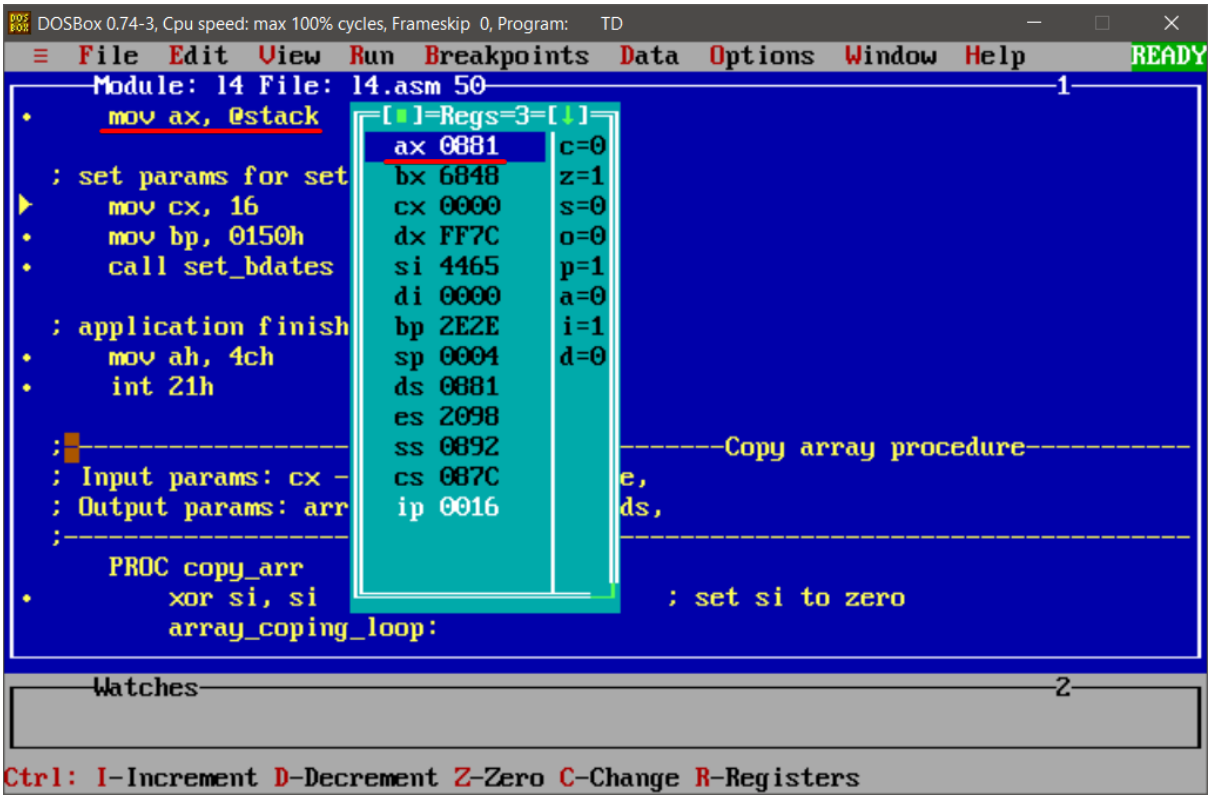
```
;----------Put array to stack procedure-----------
; Input params:  cx - initial array size,
; Output params: array copy placed at ds,
;-------------------------------------------------
    PROC array_to_stack
        xor si, si                      ; set si to zero
        push_to_stack:
            mov ax,[ds:[si]]            ; set value to ax from ds with offset si
            push ax                     ; push ax value to stack
            add si, 2                   ; si + 2
            loop push_to_stack

        ret
    ENDP
```

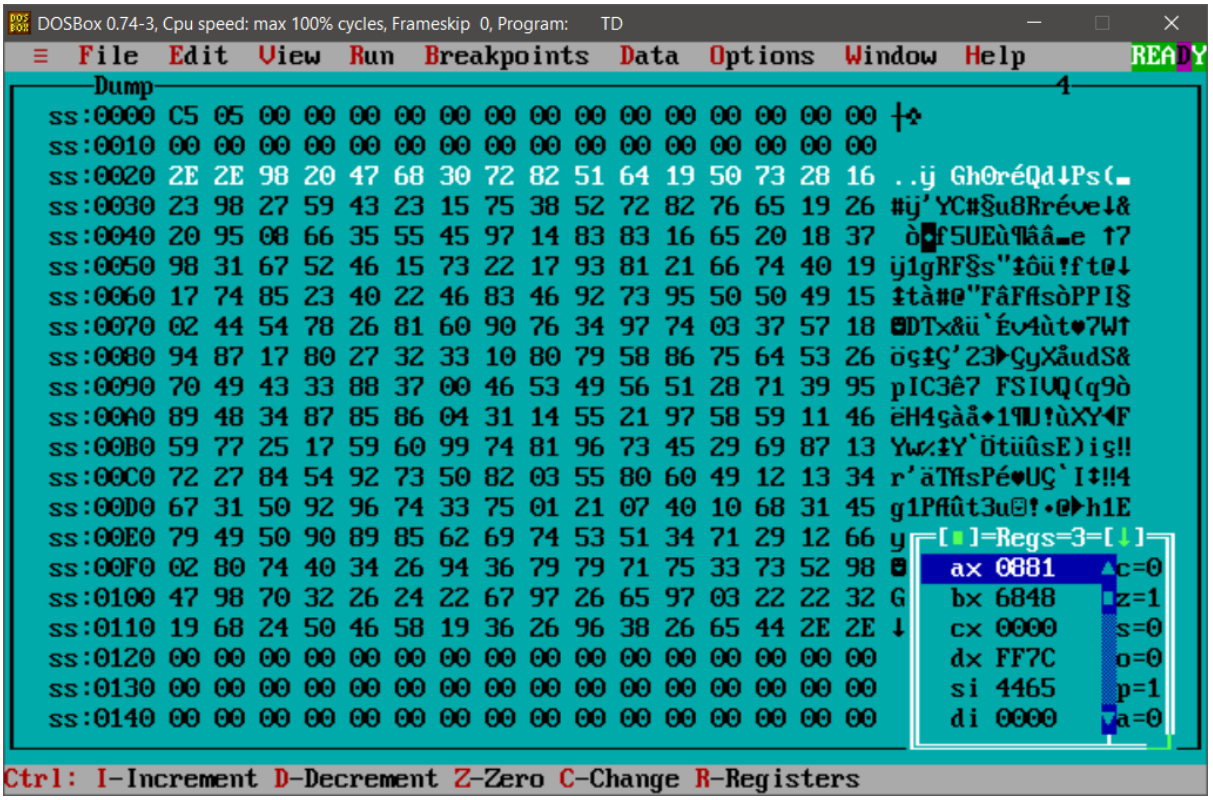
А потім змінили значення в стеці використавши `set_bdates`.

```
;------------Add birthdate to stack------------
; Input params: cx - symbols amount at birthday,
;                    bp - offset
; Output params: array copy placed at ds,
;-------------------------------------------------
    PROC set_bdates
        xor si,si                               ; set si to zero
        birthdate_label:
            mov ah, [first_birthdate+si]
            mov [bp], ah                        ; add value to stack
            inc si                              ; increment si
            inc bp                              ; increment bp
            loop birthdate_label

        ret
    ENDP
```


DOSBox 0.74-3, Cpu speed: max 100% cycles, Frameskip 0, Program: TD
File Edit View Run Breakpoints Data Options Window Help
READY
CPU 80486
ss:0000 C505 lds ax,[di]
ss:0002 0000 add [bx+si],al
ss:0004 0000 add [bx+si],al
ss:0006 0000 add [bx+si],al
ss:0008 0000 add [bx+si],al
ss:000A 0000 add [bx+si],al
ss:000C 0000 add [bx+si],al
ss:000E 0000 add [bx+si],al
ss:0010 0000 add [bx+si],al
ss:0012 0000 add [bx+si],al
ss:0014 0000 add [bx+si],al
ss:0016 0000 add [bx+si],al
ss:0018 0000 add [bx+si],al
ss:001A 0000 add [bx+si],al
ss:001C 0000 add [bx+si],al
ax 3402
bx 6848
cx 0000
dx FF7C
si 0010
di 0000
bp 0160
sp 0004
ds 0881
es 2098
ss 0892
cs 087C
ip 001C
c=0
z=0
s=0
o=0
p=1
a=1
i=1
d=0
ss:014C 00 00 00 00 38 2E 31 30 8.10
ss:0154 7C 30 39 2E 31 30 7C 32 ¦09.10¦2
ss:015C 34 2E 30 34 23 98 27 59 4.04#ÿ'Y
ss:0164 43 23 15 75 38 52 72 82 C#§u8Rré
ss:016C 76 65 19 26 20 95 08 66 ve↓& ò◘f
ss:014C 0000
ss:014A 0000
ss:0148 0000
ss:0146 0000
ss:0144 0000
Ctrl: G-Goto S-Search N-Next C-Change F-Follow P-Previous D-Display B-Block


DOSBox 0.74-3, Cpu speed: max 100% cycles, Frameskip 0, Program: TD
File Edit View Run Breakpoints Data Options Window Help
READY
Dump
ds:0240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
ds:0250 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
ds:0260 38 2E 31 30 7C 30 39 2E 31 30 7C 32 34 2E 30 34 8.10¦09.10¦24.04
ds:0270 23 98 27 59 43 23 15 75 38 52 72 82 76 65 19 26 #ÿ'YC#§u8Rrévе↓&
ds:0280 20 95 08 66 35 55 45 97 14 83 83 16 65 20 18 37 ò◘f5UEù¶ââ▬e ↑7
ds:0290 98 31 67 52 46 15 73 22 17 93 81 21 66 74 40 19 ÿ1gRF§s"‡ôü!ft@↓
ds:02A0 17 74 85 23 40 22 46 83 46 92 73 95 50 50 49 15 ‡tà#@"FâFfsòPPI§
ds:02B0 02 44 54 78 26 81 60 90 76 34 97 74 03 37 57 18 ☻DTx&ü`Év4ùt♥7W↑
ds:02C0 94 87 17 80 27 32 33 10 80 79 58 86 75 64 53 26 öç‡Ç'23►ÇyXåudS&
ds:02D0 70 49 43 33 88 37 00 46 53 49 56 51 28 71 39 95 pIC3ê7 FSIVQ(q9ò
ds:02E0 89 48 34 87 85 86 04 31 14 55 21 97 58 59 11 46 ëH4çàå♦1¶U!ùXY◄F
ds:02F0 59 77 25 17 59 60 99 74 81 96 73 45 29 69 87 13 Yw%‡Y`Ötüûs E)iç!!
ds:0300 72 27 84 54 92 73 50 82 03 55 80 60 49 12 13 34 r'äTfsPé♥UÇ`I‡!!4
ds:0310 67 31 50 92 96 74 33 75 01 21 07 40 10 68 31 45 g1Pfût3u☺!•@►h1E
ds:0320 79 49 50 90 89 85 62 69 74 53 51 34 71 29 12 66 yIPÉëàbitSQ4q)‡f
ds:0330 02 80 74 40 34 26 94 36 79 79 71 75 33 73 52 98 ☻Çt@4&ö6yyqu3sRÿ
ds:0340 47 98 70 32 26 24 22 67 97 26 65 97 03 22 22 32 Gÿp2&$"gù&eù♥""2
ds:0350 19 68 24 50 46 58 19 36 26 96 38 26 65 44 2E 2E ↓h$PFX↓6&û8&eD..
ds:0360 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
ds:0370 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
ds:0380 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Ctrl: G-Goto S-Search N-Next C-Change F-Follow P-Previous D-Display B-Block





Скориставшись командою @stack ми знайшли логічне зміщення регістру. його адреса 0881. Далі знаючи це можна розрахувати логічні та фізичні адреси 1 і останнього елемента. 1 елемент має зміщення 0020 відносно

початку стеку, а отже логічна адреса дорівнює ss:0020. а фізична — (0881 * 16) + [ss] = 08810 + 0020 = 8830. Для останнього: логічна адреса = 011Fh, а фізична — 0892Fh.

Висновок: У ході лабораторної роботи попрацювали з різними видами адресації, створили двовимірний масив і розібралися як працює найшвидший спосіб роботи з даними: стек. Також на реальному прикладі побачили як виглядає код на Асемблері написаний у процедурному стилі і які переваги і недоліки він має у порівнянні з іншими стилями написання.